

Roles as Agent Dynamics in Multi-Agent Systems

Haibin Zhu, *Senior Member, IEEE*

Department of Computer Science and Mathematics, Nipissing University
100 College Drive, North Bay, Ontario, P1B 8L7, Canada
<http://www.nipissingu.ca/faculty/haibinz/>
Email: haibinz@nipissingu.ca

Abstract: Multi-agent systems have been developed for many years. However, little attention has been paid to agent dynamics - the driving forces governing agents' lives, work and contributions to a multi-agent system. Many questions related to the dynamic properties of a multi-agent system require study. This paper investigates this concern and proposes that roles and role-based architecture could be taken as dynamics for agents in a system. This paper contributes to multi-agent systems with a potential solution to such an important challenge, i.e., agent dynamics.

Keywords: Agents, Multi-agent systems, Roles, Dynamics.

1. Introduction

Everything exists in the world for a special reason. Every activity in the world is instigated for some reason. These reasons form driving forces for new things to be created and for new actions to be taken.

In the society, people in an organization with good dynamics will work actively and collaboratively toward the common goal of the organization. In contradiction, people in an organization without good dynamics cannot work as effectively and may not have a clear understanding of the goals necessary to make the organization competitive.

In the business world, a person is required to have new knowledge, skills and habits to be qualified for a new position. Successfully finding a new job is dependent on similarities between new roles and those previously performed by the person [13].

It is the same in a multi-agent system. Multi-agent systems should encourage diversity of behavior in a population of cooperating agents [12]. This is one kind of requirement for agents. There are definite reasons for the addition of new agents to the system.

Multi-agent systems have also been investigated for many years. To build a multi-agent system, the following questions should be answered:

- How are agents created?
- *Why are agents created?*
- *Why are agents transferred on the networks?*
- *How are agents made pro-active?*

Most contributions discuss how agents are created and designed [9, 10, 17, 20, 25]. However, there is a lack of attention to the dynamics of agents in such systems [7, 19]. This is an important oversight in the research of multi-agent systems. From Physics, dynamics is the first-order problem to consider and to solve in order to model a physical world. In order to make multi-agent systems really applicable to

artificial intelligence (AI) or information technology (IT), this problem must be addressed. There must be some motivation for agents to join, work and advance in a system.

In this paper, roles are considered as the major source of dynamics for agents. To build a multi-agent system, roles can be taken as nodes in the distributed structure or architecture. Agents are placed on this structure or architecture to make the system work properly. Roles are the reasons why agents are created, modified, and transferred. Roles are also a tool to make agents pro-active. An agent should be able to find roles they wish to perform as demonstration of its ability and mobility.

This paper is organized as follows. Section 2 reviews the application of roles in agent systems; Section 3 describes the revised E-CARGO model, Section 4 discusses the environments of agents; Section 5 illustrates why agents are created; Section 6 discusses why agents are mobile; and Section 7 demonstrates how agents are made pro-active; and Section 8 concludes the paper and proposes future research.

2. Roles in Agent Systems

Many agent systems apply different aspects of the role mechanisms. The role concepts are generally used in agent systems to describe the collaboration among cooperative processes or agents.

Betch et al. pioneered consideration of roles as interfaces and developed a multi-agent system ROPE [2]. The cooperative processes are described with roles. An agent that wants to take part in the cooperation must fulfill the service requirement specified by a role. A role is formally defined as an entity consisting of a set of required permissions, a set of granted permissions, a directed graph of service invocations, and a state visible to the runtime environment but not to other agents.

Taking roles as specific processes is the main idea of Stone and Veloso's soccer team formation [18]. A role consists of a specification of an agent's internal and external behavior. They use roles to obtain a flexible team agent structure and a method for inter-agent communication. Agents can flexibly switch roles within formations and change formations dynamically, according to predefined triggers to be evaluated at run-time. This flexibility increases the performance of the overall team.

In agent-oriented software engineering, Gaia methodology [23, 24] is proposed to support system analysis and design by taking a multi-agent system as an organization. In Gaia, roles are described with responsibilities, permissions, interaction

protocols and activities. The ideas applying roles in agent-oriented software engineering are similar to those in our previous work [29, 30]. This paper follows this idea.

In Organization-Centered Multi-Agent Systems (OCMAS) [8], roles are emphasized as an important element of organizations. A role is the abstract representation of a functional position of an agent in a group. An agent is open for the whole system; there are no constraints for agents to access others and agents interact with each other directly.

Cabri et al. [5-6] apply both aspects of agent-roles: processes and interfaces. A role imposes a defined behavior on the objects playing it, thereby reflecting the process aspect. It also allows a set of capabilities, which can be exploited by agents to carry out their tasks, reflecting the interface aspect. This allows a role to serve as an abstract description for the functions an agent is responsible to fulfill in order to reach an assigned goal. From an agent-oriented approach, roles are a proper means of refining agent-oriented models. Their work [5-6] considers rights as part of their agent-roles.

Based on the work of such psychologists as Biddle and Thomas [3] and Shakespeare's role concepts in theaters, Odell et al. [14, 15] specify roles, with both the aspects mentioned above, as a superstructure specification that defines the user-level constructs required to model agents and their groups. Roles specify normative behavioral repertoires for agents and provide both the building blocks for social agents and the requirements by which agents interact.

Partsakoulakis and Vouros review the role concepts and mechanisms in multi-agent systems [16]. Roles are viewed as tools to manage the complexity of tasks and environments. High degree of interactions, environment dynamics and distributivity are emphasized to show the importance of roles. They use five aspects to analyze the past research on agent-roles: specification, dynamics of assignments, dynamics of roles, cardinality of roles and life-span of roles.

In agent systems, the following consensus is reached [5-6, 14-19, 23, 24]:

- (i) Roles can be used to specify interactions.
- (ii) Roles are agent-building blocks in class hierarchies.
- (iii) Roles can express the organizational structure of a multi-agent system.
- (iv) A role instance is deleted when an agent is destroyed, i.e., its lifetime depends on its agents.
- (v) A role can be either taken as an interface or a process.
- (vi) Interface-roles are used to form different interfaces for agents in order to restrict the visibility of features and to handle permissions for the access to the internal state and role services of agents.
- (vii) Process-roles specify how the players should behave. They have three functions: comprise special behavior, form the behavior of an agent, and take a position in a group of agents.

3. A Revised E-CARGO Model

In traditional agent systems with roles, roles are normally used as interaction media for agents. Most of them ignore the functions of roles as dynamics. In the following discussion, the interface-role concepts are used [28].

To support role-based collaboration, the E-CARGO model is proposed [29] and adopted to encourage people to contribute in collaboration [27]. A multi-agent system Σ can

be described as a 8-tuple $\Sigma ::= \langle C, O, \mathcal{A}, \mathcal{M}, \mathcal{R}, \mathcal{E}, \mathcal{G}, s_0 \rangle$, where C is a set of classes, O is a set of objects, \mathcal{A} is a set of agents, \mathcal{M} is a set of messages, \mathcal{R} is a set of roles, \mathcal{E} is a set of environments, \mathcal{G} is a set of groups, s_0 is the initial state of a collaborative system.

An *object* is *data* to be accessed by the *agents*. A *class* is a template for a group of similar objects to express their *data structures* and *operations* that are mainly access operations to the data such as read and write.

A *message* is defined as $m ::= \langle n, v, l, \mathcal{P}, t, w \rangle$ where n is the identification of the message, v is null or the receiver of the message expressed by an identification of a role, l is the pattern of a message, specifying the types, sequence and number of parameters, \mathcal{P} is a set of objects taken as parameters with the message pattern l where $\mathcal{P} \subset O$, t is a tag that expresses any, some or all message, w is the weight for an agent to collect its credit for promotion, i.e., if an agent or its human user responds this message the agent will get the weight and accumulate it to its credit.

A *role* $r ::= \langle n, I, \mathcal{N}_a, \mathcal{N}_o, p, \mathcal{R}_m, \mathcal{R}_s, w \rangle$ where, n is the identification of the role, $I ::= \langle \mathcal{M}_{in}, \mathcal{M}_{out} \rangle$ denotes a set of messages, wherein, \mathcal{M}_{in} expresses the methods or functions to respond the incoming messages, \mathcal{M}_{out} expresses a set of outgoing messages or message templates to roles, \mathcal{N}_a is a set of agents that are playing this role; and \mathcal{N}_o is a set of objects that can be accessed by the agents playing this role, p is used to express the ability requirement, \mathcal{R}_m the super roles, \mathcal{R}_s the subordinate roles, and w the credit requirement. Note that, w and p are extra criteria to check if an agent is able to play the role besides that the agent's services cover the \mathcal{M}_{in} and the requests are covered by the \mathcal{M}_{out} .

An *environment* expresses a structure to build a group. It is specified by roles, the objects accessed by the roles and the cardinalities of agents playing roles (See Section 4 for details).

An *agent* is a special object with the properties of agents. It is defined as $a ::= \langle n, c_a, s, r_c, \mathcal{R}_p, \mathcal{N}_g, p, w, t_i \rangle$, where, c_a is a special class that describes the common properties of users, n is the identification or name of the agent, s is the set of properties of the agent, r_c is called *the current role* and means a role that the agent is currently playing, \mathcal{R}_p means a set of roles (called *active roles*) that the agent is potential to play ($r_c \notin a.\mathcal{R}_p$), and \mathcal{N}_g means a set of groups the agent belongs to; p is used to express the ability, w the credits, and t_i the idle time.

A *group* is a set of agents that are working in an environment, i.e., a set of agents assigned with roles in the relevant environment.

By E-CARGO, a multi-agent system is built and activated by equipping agents with the following components:

- Objects are classified and instantiated for the system.
- Roles are specified with the requirement of the cooperation of the system.
- Agents are designed with procedures corresponding to the roles they play.
- An environment with associated roles and objects are specified;
- Agents form a group by playing roles in an environment.
- Agents autonomously pursue the goals regulated by playing current roles, transferring active roles,

pursuing future roles to cooperate with each other and reach their common goal.

4. Environment

To live in the world is to adapt to the environment. Therefore, environments significantly affect a person's development. Similarly, environments affect the design of intelligent agents. Environments are complex and complicated. Russell and Norvig [18] investigate the nature of environments for agents. An environment might be:

- Fully observable or partially observable;
- Deterministic or stochastic;
- Episodic or sequential;
- Static or dynamic; and
- Discrete or continuous.

It is our intention to model and abstract it into a controllable entity. Roles can be mechanisms to form environments for agents to play in a multi-agent world, because roles can express rationalities, requirements, and dynamics.

In E-CARGO, environments are the key component to represent the dynamic properties of a multi-agent system. An environment $e ::= \langle n, \mathcal{B} \rangle$ where, n is the identification of the environment and \mathcal{B} is a set of tuples of role, number range and an object set, $\mathcal{B} = \{ \langle r, q, \mathcal{N}_o \rangle \}$. The number range q tells how many users may play this role in this environment and q is expressed by $[l, u]$, and $[l]$ is used when $l = u$. For example, q might be $[1]$, $[2]$, $[1, 10]$, $[3, 50]$, It states that how many agents may play the same role r in the group. The object set \mathcal{N}_o expresses the complex objects accessed by the agents who play the relevant role. "Complex" means that they are composed of other objects. The complex objects in \mathcal{N}_o are categorized as two kinds: singly owned or shared, i.e., one complex object in this set may be accessed by one agent or shared by many. In fact, \mathcal{N}_o expresses the resources for agents to access.

With the above discussion, the agents in E-CARGO are fully consistent with the idea that autonomous agents are situated in some environment emphasized by Franklin and Graesser [9] although they do not define what an environment is.

For example, a soccer team simulation with the 4-4-2 formulation as an environment as

- $e_1 = \{ \langle \text{goalie}, [1], f \rangle, \langle \text{defender}, [4], f \rangle, \langle \text{midfield}, [4], f \rangle, \langle \text{forward}, [2], f \rangle, \langle \text{goalie}, [1], f \rangle \}$, where, $f = \langle \text{the gate, the field, the ball} \rangle$.

A 3-6-1 formulation can be described as

- $e_2 = \{ \langle \text{defender}, [3], f \rangle, \langle \text{midfield}, [6], f \rangle, \langle \text{forward}, [1], f \rangle, \langle \text{goalie}, [1], f \rangle \}$.

If the team is described more specifically, the environments are:

- $e_1 = \{ \langle \text{goalie}, [1], f \rangle, \langle \text{left-defender}, [1], f \rangle, \langle \text{right-defender}, [1], f \rangle, \langle \text{mid-defender}, [2], f \rangle, \langle \text{left-midfield}, [1], f \rangle, \langle \text{right-midfield}, [1], f \rangle, \langle \text{mid-midfield}, [2], f \rangle, \langle \text{left-forward}, [1], f \rangle, \langle \text{right-forward}, [1], f \rangle, \langle \text{goalie}, [1], f \rangle \}$.
- $e_2 = \{ \langle \text{left-defender}, [1], f \rangle, \langle \text{right-defender}, [1], f \rangle, \langle \text{mid-defender}, [1], f \rangle, \langle \text{left-midfield}, [2], f \rangle, \langle \text{right-midfield}, [2], f \rangle, \langle \text{mid-midfield}, [2], f \rangle, \langle \text{left-forward}, [1], f \rangle, \langle \text{goalie}, [1], f \rangle \}$.

The environment e_i can be shown as in Fig. 1, where, R_1 - R_4 are defender roles, R_5 - R_8 are midfielder roles, R_9 - R_{10} are forward roles, and R_0 is the goalie roles. Fig. 1 also shows some relationships among roles. For example, R_0 can request R_1 , R_2 , R_3 , and R_4 ; R_2 can request R_5 and R_6 ; and R_3 can request R_7 and R_8 .

From these discussions, the environment e in the model E-CARGO is a good abstraction for the real environment of agents (Fig. 1). In E-CARGO, an environment is a design of a system. With an environment, the consistency of a system design can be evaluated. A basic rule is that for each request of a role there should be at least one role that provides the services corresponding to it; for each service of a role, there should be a role requesting it.

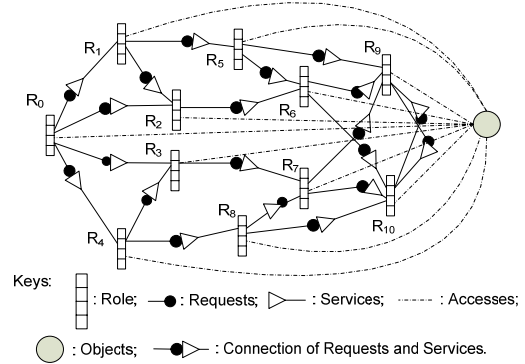


Fig. 1. A soccer agent team with the 4-4-2 formation.

5. Why Are Agents Created?

Agent dynamics is defined as a force or impetus for agents to be created, advance, progress and work in an environment, a community of agents or a system. From this point of view, agent dynamics is considered an important property of both a multi-agent system and its agents.

An agent is viewed as anything that perceives its environment through sensors and acting upon that environment through actuators [17]. A rational agent is one that does the right things. "For each possible percept sequence, it should select an action that is expected to maximize its performance measure, given the evidence provided by the percept sequence and whatever built-in knowledge the agent has [17]." Rationality is one of the dynamics of agent performance in a multi-agent system. However, the expression of rationality is not comprehensively investigated.

Roles are, in fact, the rationalities for agents to live. Computer systems are added to the Internet so as to perform certain roles and to provide services on the Internet. To design a system, role factories (Fig.2) are making roles to match the requests and services in the role pool. If there are no more empty requests and services, the role factories should cease role production. There may be a period of time when roles are not claimed by agents. After this period, role production should also stop.

The creators of agents are called agent factories. Agent factories are in fact a software development team. In this style, traditional software engineering might be significantly changed. It definitely reflects the principles of Agile Programming (AP) [21] or Extreme Programming (XP) [1].

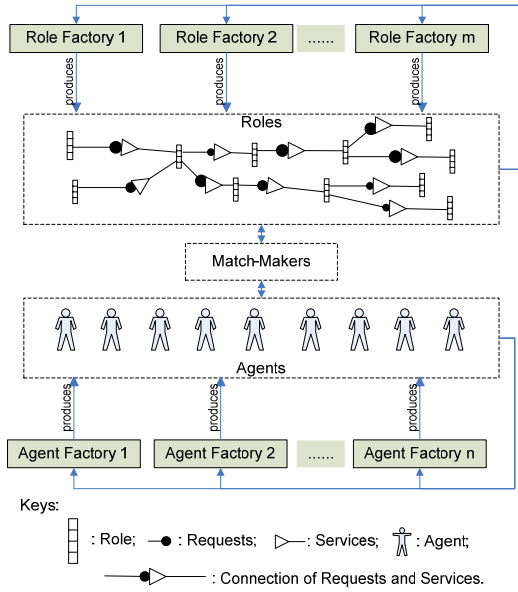


Fig. 2. An agent system with a match-maker.

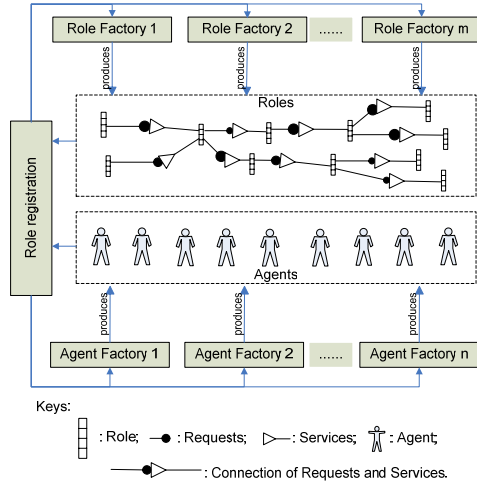


Fig. 3. An agent system with a role registration.

In role-based architecture, an agent enters the system by matching the requests and services of roles. An agent matching a role should acquire and provide the required services. The match makers know the roles and their related vacancies. In Fig. 2, matchmaking is the process that brings requester and service provider agents together [30]. Note that objects are omitted in Fig. 2 and Fig. 3 to save space. Role factories establish requirements and agent factories produce agents to satisfy these requirements. In this model, agent factories have limited information regarding roles. Therefore, the agent factories may produce agents that have no roles to play. In this case, it is a waste of agent production resources.

In Fig. 3, a role registration is used. All roles are registered. Agent factories are provided with role specifications from this process and produce agents on demand. This model is more efficient than that illustrated in Fig. 2.

A theory of dynamics should indicate what produces forces, the relationship among these driving forces, and impacts on environments and agents. The principle of batteries can be used as an analogue. Roles are cells and agents act as electrons [4]. In a battery, the power is stored when cells are charged or recharged. The power is used when

the electrons move to fill the cells. When all the cells are filled by electrons, the power disappears.

It is similar in agent systems; roles attract agents. The larger the number of roles, the greater the attraction on agents. Consider a factory that produces agents. The production of agents will be relative to the attraction by roles.

The initial force \mathcal{F} to produce an agent a intending to play role r in an environment e is proportional to the cardinalities v ($v = \frac{q.l + q.u}{2}$ is the average of the lower and upper

cardinalities) of r . It is expressed as

$$\mathcal{F} = c \times v,$$

$$\text{where, } v = \frac{q.l + q.u}{2}, \exists r \in \langle r, q, N_o \rangle \in e.B \wedge \text{can-play}(a, r),$$

c is a constant and predicate $\text{can-play}(a, r)$ means agent a is able to play role r .

The cardinality of a role is a magnitude of the force driving agent production. Driven by this force, the factory produces agents to adopt the role. This is why American universities contributed many computer science graduates to the IT market in the years 1998-2002.

Periodically, agent factories may produce a surplus of agents. Suppose for an environment e , the agent set is \mathcal{A} and the total number of the upper cardinalities of related roles is v_r , i.e., $v_r = \sum_r q.u$, where, $\langle r, q, N_o \rangle \in e.B \wedge a \in \mathcal{A} \wedge$

$\text{can-play}(a, r)$. If $|\mathcal{A}| - v_r > 0$, then the agent factories should stop creating this kind of agent. This is why the enrollment of computer science students dropped down in the years 2002-2006.

This differs from the battery analogy, in that agent production has inertia in the production pipeline. This is why many computer science graduates in the United States could not find jobs in the years 2002-2005. In multi-agent systems, this inertia or surplus should be minimized. The possibility of minimizing such inertia is due to agent copying and the short duration of new services loading.

6. Why Are Agents Transferred?

Mobility is another dynamic property of agents in multi-agent systems. It is needed to clarify why agents are transferred. The basic idea behind agent mobility in distributed computing is to conserve network bandwidth, i.e., due to their relative sizes, agents, not data, are transferred [26].

The reasons why agents are transferred across the network can also be rationalized by roles:

- (i) Agents may be light-weight or heavy weight. Mobile agents should be light-weight and are proposed as a way to save the load of transformation across the network connecting computers. This can be expressed by some components of the E-CARGO model. If $|a| < |\mathcal{N}_o|$, where a is an agent and \mathcal{N}_o the object accessed by the agent to play a role in a host, $|x|$ is used to express the size of memory allocated to x , then the agent must be mobile.
- (ii) Agents are temporary to the system that adopts them. They are not inherent parts of the system. They reside in a system for limited periods. For example, a database system is inherently a data-centered system. Many roles are attached to the system to restrict agent

accessibility. Mobile agents are used to maintain or modify data. In this sense, a database system is actually a set of data objects and roles, that is, an environment of the E-CARGO model.

- (iii) Even though software agents are much easier to replicate than their human counterparts, they cannot maintain their state in a new environment. It is well-accepted that consistent agents are always easy to be maintained, especially dynamically maintained. To facilitate cooperation in a multi-agent system, state consistency is an important agent property. Within E-CARGO, when an agent a plays a role, it accesses the data objects related to that role, and modifies its own memory to keep its current state s , collects its credit w , expand its abilities p_i and adjust its workload d . This means that an agent always keeps its state consistent and maintained by itself.
- (iv) Agents may be idle due to a lack of roles or service requests. Agents must be reclaimed by the agent factories and rebuilt to adopt more or new roles. In E-CARGO, for an agent a , if $a.t_i$ is larger than a limit, it should be reclaimed. In the factories, agents are loaded with new capabilities and capacities then returned to the environment.

7. How Are Agents Made Pro-active?

Pro-activeness is a necessary condition for being an agent. This is a mandatory property of agent based systems [23]. Agents are generally processes created from pre-designed programs. In traditional ways, many rules must be designed in the programs to specify their activities. This method is definitely not a good simulation of human behavior.

Agent dynamics are based on their adaptabilities [10]. Dynamics generate motivational or driving forces, physical or moral, in any field. The human world follows the Maslow's hierarchy [11] of needs. Agents can also be designed in this way.

The assumptions for dynamic role adoption are based on this hierarchy of needs. Three kinds of agents can be classified [27]:

- (i) Diligent agents try to do as many jobs as possible. They seek as many responsibilities as possible without concern for rights.
- (ii) Ordinary agents seek only to complete jobs assigned to them. They take responsibilities but require matching rights.
- (iii) Lazy agents try to accomplish a minimum number of tasks. They seek a maximum number of rights with minimal responsibility.

To restrain type (iii), encourage type (i), and support type (ii), roles should be composed of both rights and responsibilities. That is to say, additional rights demand increased responsibilities. If agents hope to grasp more rights, they must adopt more roles. As a result, they must take on additional responsibilities. For those who do not care about rights, there would be still enough rights for them to take additional responsibilities. This is the fundamental principle in role-based collaboration [29, 30].

With this fundamental principle, role mechanisms can be taken as dynamics for agents to be pro-active and mobile in an environment. By adopting roles, agents can progress by

telling how many significant roles they are playing. Roles can be taken as a flag of advances for agents in collaboration.

In a role-based multi-agent system, lazy agents will be finally discarded and reclaimed by the agent factories. Diligent agents will occupy the majority of the system and be promoted to play more important roles providing them opportunities for greater contribution. In E-CARGO:

- Roles can be specified with credits (w) and services (M_i).
- Agents can earn credits (w) by playing roles.
- Agents have abilities (p_i) to match the ability requirement of a role (p_i) to provide services to play a role.
- Agents are trained (to load more service implementations) to match the service requirement of the role (M_i) by paying an amount of time by decreasing an amount of its ability (p_i).
- The goal of agents is to earn a maximum number of credits (w).
- To obtain the goal, agents must compete, coordinate and collaborate.
- The goal for collaboration is to collect more credits.

From the above assumption, an agent can be set a goal with definite credits and then started. The agents will automatically search out roles and adopt them.

After agents become pro-active to pursue roles, a new problem may occur. A multi-agent system should have maximum cooperation and minimum competition. If agents are designed to adopt roles and collect as many credits as possible, there must be competitions among agents. This may introduce negative factors into the system. Fortunately, agents are designed by people and they cannot grow spontaneously. The role distributions can be even and fair based on the abilities of agents and distances between agents and roles. If an agent loses a bid for a role, it should seek out others. If it is inactive for more than a specified period of time, it should be reclaimed by the factories and rebuilt. Competition can be avoided in this way.

8. Conclusion

In this paper, three main questions about agents are answered: (1) Why are agents created? (2) Why are agents transferred? (3) How are agents made pro-active? The major answers are related with roles. Therefore, roles should be fully considered and specified at first before a multi-agent system is built.

In agent system implementations, roles are abstract and agents are concrete. Roles are good guidelines for agent design and implementation. Roles are also hints for agents to show their dynamic properties.

Representing capabilities of agents is difficult. The dynamic properties of agents can be expressed with roles. Effective role specification, agent and role matching algorithms and structures, and related agents' abilities are important research topics that require additional comprehensive research.

Acknowledgement

This research is supported in part by NSERC (National Science and Engineering Research Council, Canada, No. 262075-06) and the IBM Eclipse Innovation Grant Funding.

Thanks also go to Mike Brewes of Nipissing University for his proofreading this article.

References

- [1] K Beck, Embracing change with extreme programming, *IEEE Computer*, Oct 1999, vol. 32, no. 10, pp. 70-77.
- [2] M Becht, T Gurzki, J Klarmann and M Muscholl, ROPE: Role oriented programming environment for multiagent systems, *Proc. of Fourth IECIS International Conference on Cooperative Information Systems*, Sept. 1999, pp. 325-333.
- [3] B J Biddle and E J Thomas (Ed), *Role Theory: Concepts and Research*, John Wiley & Sons, Inc., 1966.
- [4] K Buckle, How do batteries store and discharge electricity? *Scientific American .com*, avail: http://www.sciam.com/askexpert_question.cfm?articleID=0001A09D-0D79-1476-8D7983414B7F0000&catID=3&topicID=4, 2006.
- [5] G Cabri, L Ferrari and L Leonardi, Injecting roles in Java agents through runtime bytecode manipulation, *IBM Systems Journal*, vol. 44, no. 1, 2005, pp.185-208.
- [6] G Cabri, L Ferrari and L Leonardi, Supporting the development of multi-agent interactions via roles, *The 6th International Workshop on Agent-Oriented Software Engineering (AOSE) at AAMAS 2005*, Utrecht, The Netherlands, July 2005, *Springer LNCS*, vol. 3950, pp.154-166.
- [7] C Claus and C Boutilier, The dynamics of reinforcement learning in cooperative multiagent systems, *Proc. of the 15th national/tenth conference on Artificial intelligence/Innovative applications of artificial intelligence*, Madison, Wisconsin, United States, 1998, pp. 746 – 752.
- [8] J Ferber, O Gutknecht and F Michel, From agents to organizations: an organizational view of multi-agent systems, *Giorgini, P., Müller, J., and Odell, J. (Eds.), Agent-Oriented Software Engineering (AOSE) IV, Melbourne, July 2003, Lecture Notes on Computer Science*, vol. 2935, 2004, pp. 214-230.
- [9] S Franklin and A Graesser, Is it an agent, or just a program? a taxonomy for autonomous agents, *Proc. of the Workshop on Intelligent Agents III, Lecture Notes on Computer Science*; vol. 1193, 1996, pp. 21-35.
- [10] B Hayes-Roth, An architecture for adaptive intelligent systems, *Artificial Intelligence*, vol. 72, no. 1-2, Jan. 1995, pp. 329 – 365.
- [11] A H Maslow, *Motivation and Personality (2nd ed.)*, Harper & Row, 1970.
- [12] N Minar, K H Kramer and P Maes, Cooperating mobile agents for mapping networks, *Proc. of the First Hungarian National Conference on Agent Based Computation*, 1998, avail: <http://parasol-www.cs.tamu.edu/people/amato/Courses/689-608/presentations/routes-coopagents.pdf>.
- [13] N Nicholson, A Theory of Work Role Transitions, *Administrative Science Quarterly*, vol. 29, no. 2, Jun., 1984, pp. 172-191.
- [14] J Odell, M Nodine and R Levy, A metamodel for agents, roles, and groups, Odell, J., Giorgini, P., Müller, J. (Ed.), *Agent-Oriented Software Engineering (AOSE), Lecture Notes on Computer Science*, vol. 3382, Springer, Berlin, 2005, pp. 78-92.
- [15] J Odell, H van Dyke Parunak and M Fleischer, The role of roles in designing effective agent organizations, Garcia, A., Lucena, C., Zambonelli, F., Omicini, A. and Castro, J. (Eds.), *Software Engineering for Large-Scale Multi-Agent Systems, Lecture Notes on Computer Science*, vol. 2603, Springer, Berlin, 2003, pp. 27-38.
- [16] I Partsakoulakis and G Vouros, Roles in MAS: managing the complexity of tasks and environments, Wagner, T. (Ed.), *An Application Science for Multi-Agent Systems*, Springer, 2004, pp.133-154.
- [17] D Russell and P Norvig, *Artificial Intelligence: A Modern Approach (2nd Ed.)*, Pearson Education, Inc., Upper Saddle River, New Jersey, 2003.
- [18] P Stone and M Veloso, Task decomposition, dynamic role assignment, and low-bandwidth communication for real-time strategic teamwork, *Artificial Intelligence*, vol. 110, no. 2, 1999, pp. 241-273.
- [19] J M Swaminathan, S F Smith and N M Sadeh, Modeling supply chain dynamics: A multiagent approach, *Decision Sciences*, vol. 29, no. 4, Summer 1998, pp. 607-632.
- [20] K Sycara, S Widoff, M Klusch and J Lu, Larks: dynamic matchmaking among heterogeneous software agents in cyberspace, *Autonomous Agents and Multi-Agent Systems*, vol. 5, no. 2, June 2002, pp. 173-203.
- [21] D Thomas, Agile programming: design to accommodate change, *IEEE Software*, May/June 2005, vol. 22, no. 3, pp. 14-16.
- [22] M Wooldridge and N Jennings, Intelligent Agents: Theory and Practice, *The Knowledge Engineering Review*, vol. 10, no. 2, 1995, pp. 115-152.
- [23] M Wooldridge, N R Jennings and D Kinny, The Gaia methodology for agent-oriented analysis and design, *Journal of Autonomous Agents and Multi-Agent Systems*, vol. 3, no. 3, Sept. 2000, pp. 285-312.
- [24] F Zambonelli, N R Jennings and M Wooldridge, Developing multiagent systems: the Gaia methodology, *ACM Transactions on Software Engineering Methodology*, vol. 12, no. 3, July 2003, pp. 317-370.
- [25] J E White, Mobile agents, *Bradshaw, J.M., (Ed.), Software Agents*, MIT Press, 1997, pp. 437 – 472.
- [26] Z Zhang and C Zhang, An improvement to matchmaking algorithms for middle agents, *Proc. of the First International Joint Conference on Autonomous Agents and Multiagent Systems: Part 3*, 2002, Bologna, Italy, pp. 1340 – 1347.
- [27] H Zhu, Encourage participants' contributions by roles, *IEEE International Conference on Systems, Man and Cybernetics*, Oct. 2005, Hawaii, USA, pp. 1574-1579.
- [28] H Zhu, Role mechanisms in collaborative systems, *International Journal of Production Research*, vol. 41, no. 1, Jan. 2006, 181-193.
- [29] H Zhu and M C Zhou, Role-based collaboration and its kernel mechanisms, *IEEE Trans. on Systems, Man and Cybernetics, Part C*, vol. 36, no. 4, July 2006, pp. 578-589.
- [30] H Zhu and M C Zhou, Supporting software development with roles, *IEEE Trans. on Systems, Man and Cybernetics, Part A*, vol. 36, no. 6, 2006, pp. 1110-1123.